논문 2020-15-29

Dynamic Adjustment of Hello and Hold Timer in AODV Routing Protocol

Daniel Godfrey, Ki-Il Kim*

Abstract : Ad hoc On-demand Distance Vector (AODV) protocol and its variants employ two important timers, hello and hold timer to keep track of topology changes. Moreover, hold timer is computed by multiplying constant value to hello timer. But, this configuration leads to inaccurate settings of hold timer. To solve this problem, in this paper, we propose a new dynamic adjustment of hello and hold timer scheme by removing dependency between them. A new metric to measure mobility is applied into hello timer, while expected link lifetime does holder timer. Simulation results show a significant reduction in the number of messages, a fact suggesting that it is possible to maintain and in some cases improve the performance of AODV with a minimum amount of messages released into the network.

Keywords : AODV, Link duration, Link change rate, Hello timer, Hold timer

Ⅰ. Introduction

Mobile Ad-hoc Networks (MANETs) are wireless networks consisting of self-configured mobile nodes communicating with each other over a wireless channel without the need for a fixed infrastructure. Regardless of their outstanding practical advantages in the deployment of mobile networks, routing data in MANETs is still a challenging task due to technical challenges brought by dynamic network topology and limited resources such as power and bandwidth [1]. In traditional MANETs, nodes use periodic refresh messages to constantly track local link changes and update accordingly. Despite their importance, Traditional MANETs nodes tend to send refresh messages at fixed rates, of which were preconfigured based on a trial-and-error approach.

To track topology changes caused by the random movement of mobile nodes, MANET uses different techniques such as link-layer sensing and refresh messages, commonly referred to as Hello messages [2]. Hello messages provide a mechanism for neighborhood discovery and update process. The validity of topology information exchanged between nodes highly depends on two important timers; namely the Hello interval and the neighbor timeout timer. The latter determines the time during which detected neighbor information is considered to be valid [3]. Depending on the type of MANET protocol being implemented, nodes may require to regularly update and preconfigure routes to every other node in the network (proactive routing) or may do so only if required (reactive routing) [4], hence the importance of Hello message mechanism varies depending on

---

*Corresponding Author (kikim@cnu.ac.kr)

Received: Jul. 9, 2020, Revised: Aug. 25, 2020, Accepted: Sep. 1, 2020.

D. Godfrey: Chungnam National University (Ph.D. Student)

K. Kim: Chungnam National University (Prof.)

※ This research was supported by Basic Science Research Program through the National Research Foundation of Korea funded by the Ministry of Education (2018R1D1A1B07043731).

ⓒ IEMEK J. Embed. Sys. Appl. 2020 Oct. 15(5) 251-259
ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2020.15.5.251

the protocol being implemented. Regardless, this does not change the fact that, the use of pre-configured fixed Hello messages to track and update local link information is not always optimal for network performance. For example, sending Hello messages at a high rate has proven to reduce data packets delivery latency, which eventually increases network throughput [5], However, it comes at a cost of increased consumption of available bandwidth and energy as shown in the study [6].

Several studies have been done to study and observe the impacts of using different Hello message intervals and its related parameters on the performance of MANETs. In study [7], simulation results suggest that, sending Hello messages at a higher rate when mobility of nodes is high enhances network throughput and reduces neighbor detection latency. However, tuning Hello messages does not guarantee better performance in all situations as its effectiveness depends on many other factors such as node density [8].

To adaptively adjust Hello message intervals, study [9] suggests the faster the node moves, the lower its Hello interval should be so as to fasten link detection. However this approach seems to ignore important facts such as, relative movement between nodes. As the link between two nodes moving at high speed can exist as long as the movement of direcion is the same for both nodes. For that reason, speed should not be the only factor to adjust Hello intervals as shown in study [10]. The authors proposed a perspicuous approach to measure network stability by constantly tracking local topology change rates and name the parameter, link change rate (LCR). The higher the rate, the faster the node is required to send Hello messages. In this study, the LCR parameter is used to adaptively adjust Hello interval, while neighbor timeout timer just like the traditional mechanism, it is a function of neighbor Hello interval. It is clear that, updating neighbor timeout timer in this way is somehow uncertain since it does not estimate the time remained before link to neighbor is broken. Updating neighbor timeout timer in this manner leads to inaccurate estimation of local link lifetime which could lead to data loss problems such as misrouting and neighbor-flapping as discussed in study [11]. Additionally, neighbor hold timer is subject to frequent change since neighbor Hello intervals are constantly changing as well.

Take in consideration all these facts, Our study proposes a simple yet a clear approach to measure link lifetime and use it to update neighbor timeout timer and based on it we proceed to adjust Hello intervals accordingly. The proposed approach is expected to eliminate inconsistencies shown by previous studies especially when storing neighbor timeout timer values and on estimation of the parameter link change rate. We introduced an assumption that, The longer the neighbor timeout timers are, the less local topology changes are expected to occur in a given time and in such cases, larger Hello intervals are desirable.

To estimate link lifetime of a wireless link between two nodes, we utilized extended Hello messages as a means to exchange important mobility information such as locations and velocity as shown in Section II. We expect to see an adequate reduction in number of Hello messages released into the network especially in situations with low mobility. We verify the usefulness of our proposed scheme by using the network simulator ns-3.29 [12]. The rest of this paper is organized as follows. Section II present in detail on the proposed mechanisms and their application on AODV routing protocol. Section III presents simulation results and discussions in detail. We conclude our study and present future works in Section IV.

Ⅱ. Proposed Dynamic Timer Scheme

In MANETs protocols, validity of a link between two nodes depends on the parameter neighbor timeout timer which simply refers to maximum time for a node to wait for any update message from its previously detected neighbor before it considers the link to that neighbor as either lost or broken. In traditional MANET routing protocols the parameter neighbor timeout timer is preconfigured to be the maximum tolerated numbers (n) of missed Hello messages from a particular neighbor node before it is considered as lost. This important timer is defined and expressed as neighbor timeout timer = n × helloInterval. In study [13] the authors suggests that, wireless link lifetime depends on two main factors which are; the relative mobility (speed and direction) between the two nodes and their respective transmission range. As mentioned before, our study approaches the adaptive timer problem by first estimating neighbor timeout timer through information exchanged through the extended Hello messages. The adaptive timer mechanisms discussed in this study depends on the following parameters; which are; the expected link duration (ELD), the link change rate (LCR) and finally a combined approach (ELD+LCR). Further description on each scheme is given in detail below.

1. Expected Link Duration (ELD)

A link duration from a node to its neighbor refers to an expected amount of time remained before a neighbor moves at a distance greater than the its transmission range. As previously mentioned, we utilized the extended Hello messages to allow nodes to announce their individual mobility behaviour as described in the following subsection.

1.1 Exchange of mobility Information

We use the commonly applied mobility model, the Random Waypoint Mobility model to control the mobility behavior of mobile nodes in a given simulation area. With this model, nodes tend to move at a constant speed towards a randomly selected destination within the simulation area and pause for a certain period of time (pause time) before choosing another destination randomly.

<table>
<tr><td>Type</td><td>R</td><td>A</td><td>Reserved</td><td>Prefix Size</td><td>Hop Count</td></tr>
<tr><td colspan="6">Destination IP Address</td></tr>
<tr><td colspan="6">Destination Sequence Number</td></tr>
<tr><td colspan="6">Current coordinate (X)</td></tr>
<tr><td colspan="6">Current coordinate (Y)</td></tr>
<tr><td colspan="6">Destination coordinate (X)</td></tr>
<tr><td colspan="6">Destination coordinate (Y)</td></tr>
<tr><td colspan="6">Node velocity</td></tr>
</table>

Fig. 1 Modified hello message header

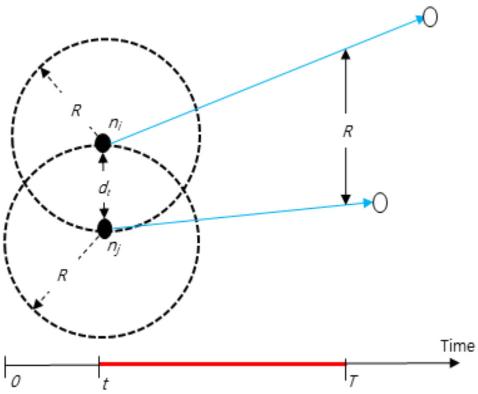

Fig. 2 Link initiation process

Using the modified Hello message header shown in Fig. 1, nodes are able to exchange mobility information (location coordinates and velocity) and compute link duration to their respective neighbors as described in the next subsection.

1.2 Computation of the ELD

From Fig. 2 below, consider two mobile nodes n_i and n_j of the same transmission range R meters. The two nodes are expected to create a new link and become neighbors at any time (say t) when they move closer to be at a distance of separation d_t (which is less than R) meters. In that case, the estimated link duration (T-t), is an estimated amount of time to be taken before the two nodes move away to be at a separation distance that is larger than their respective transmission range.

A node computes the link duration to its neighbor immediately after receiving an extended Hello message from its neighbor. We used the method proposed in the study [14] to estimate the link duration of mobile nodes moving under the impact of the random waypoint mobility model. We set bounds to limit the value of the calculated expected link duration to be between eld_max(upper bound) and eld_min(lower bound). The upper bound allow nodes to send minimum number of Hello messages even when nodes are completely static, while the lower bound limits the maximum number of Hello messages to be sent even at very high mobility conditions. As mentioned before, it is clear that the link duration between two mobile nodes depends on, their relative velocity and their respective transmission range [15].

1.3 Adaptive timer adjustment mechanism

In our work, we allow nodes to update neighbor timeout timer according to the computed expected link duration as shown in Fig. 3 below.

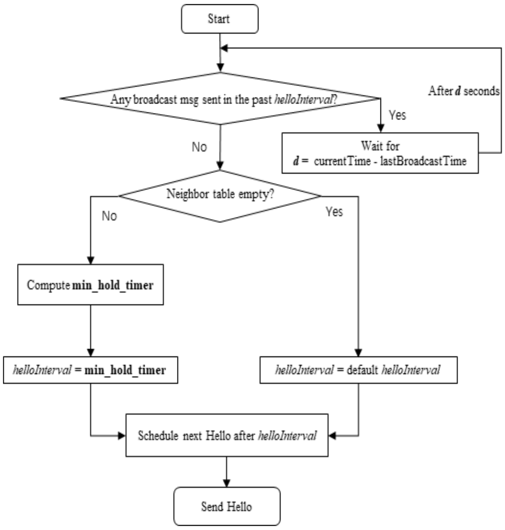

Fig. 3 ELD based Hello interval adjustment scheme

Before sending the next Hello message, a node first check if any broadcast message was sent during the past helloInterval seconds. If none was sent and its current neighbor table is empty then the next Hello message will be sent at an interval of 1 second (default value)to speed up neighbor discovery process. If not empty then the next Hello will be scheduled and sent after ***min_hold_timer*** which is the lowest stored neighbor timeout value in milliseconds. The ***min_hold_timer*** is set to not less than the ***eld_min***.

2. Combined Scheme (ELD+LCR)

To adaptively adjust Hello interval based on measured link change rate (LCR), we use similar approach as one proposed in study [10]. This study defines the parameter link change rate as the number of local links change per time. In addition to the link change rate parameter, authors proposed an additional parameter “packets drop count” for immediate detection of a link failure. We do not apply this additional method as we intend to observe the effectiveness of the parameter LCR when compared to the ones we propose. The parameter link change rate is calculated using the expression:

$$lcr = \frac{LostLinks_{counter} + DetectedLinks_{counter}}{t}.$$

where by t is the time passed since the last measurements were taken. Other parameters were left unaltered as described in study [10].

It is clear that, the number of $LostLinks_{\text{counts}}$ depend on how we update and store detected neighbor timeout timers. For the best of our knowledge, previous adaptive timer schemes have stored this value as a function of neighbor node Hello interval, which is clearly not an accurate way to measure the expected *neighbor timeout timer*. We use extended Hello messages to estimate link duration and apply it when updating neighbor timeout timer hence guaranteeing a more correct estimation of

Table 1. Simulation parameters

Parameter	Value used
Propagation delay	Constant speed propagation model
Propagation loss	Friis propagation loss model
MAC protocol	IEEE 802.11
Pause time (s)	1
Simulation time (s)	300
Transmission range (m)	250
Traffic model	CBR sources
Node speed (m/s)	10, 20, 30
Number of nodes	20
Simulation area (m^2)	500 x 500
Packet size (bytes)	512
Traffic rate (Kbps)	10
eld_max (s)	4
eld_min (s)	0.5

expiredLinks counts when estimating the link change rate and hence provide a better adaptive mechanism when adjusting the Hello interval using the new modified LCR of which we named the ELD+LCR mechanism. Side to side of all the presented mechanisms is given and discussed below.

III. Performance Evaluation

As previously mentioned, we use four different versions of the AODV routing protocol, that is, original AODV with static time, AODV with ELD, AODV with LCR as in study [10] and finally the AODV with combined ELD and LCR, to compare and discuss performance evaluation. Each scheme is represented as AODV, ELD, LCR and ELD+LCR, respectively in all figures, from Figure 4 to 7.

During simulations, a total of 10 different sets of randomly generated scenarios are used to run the traditional AODV approach and later exactly the same scenarios are used to test the other adaptive timer schemes. An average from all the ten sets is taken to give the overall results. Five Constant Bit Rate (CBR) source nodes are allowed to generate data packets of 512 bytes at a rate of 10Kbps 200~201 seconds after simulation begins for a period of 90 seconds. Configuration of all the important parameters is presented and summarized in Table 1.

1. Performance Measuring Parameters

For performance comparison, we used four performance measuring parameters which are: overhead due to Hello messages sent, data packets delivery ratio, normalized routing overhead and the end-to-end delay. Number of hello messages sent counts for the average number of total Hello messages sent by all nodes during simulation time. Packet delivery rate depends on the ability of the scheme to discover, repair and maintain detected links. The proposed adaptive schemes are expected to enhance ability of nodes to track mobility changes and react accordingly. In addition, normalized routing overhead measures an average number of Hello messages required to both initiate and update links to achieve successful deliver of a single data packet to its desired destination. Finally, end-to-end delay measures the average of time taken for a data packet takes to transfer across the network from source to destination node.

2. Performance analysis of Discussed Schemes

We observe how the four discussed schemes react to mobility change of random moving nodes. We create three scenarios to observe their effectiveness, the low mobility (10 m/s), moderate mobility (20 m/s) and high mobility (30 m/s) while maintaining other simulation parameters throughout simulations.

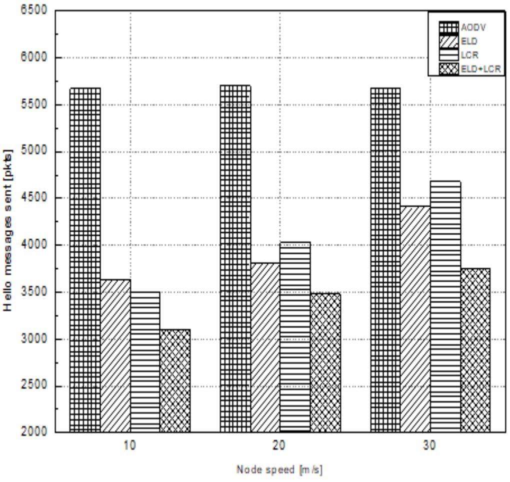

Fig. 4. Overhead in a network of 20 nodes as a function of varying mobility

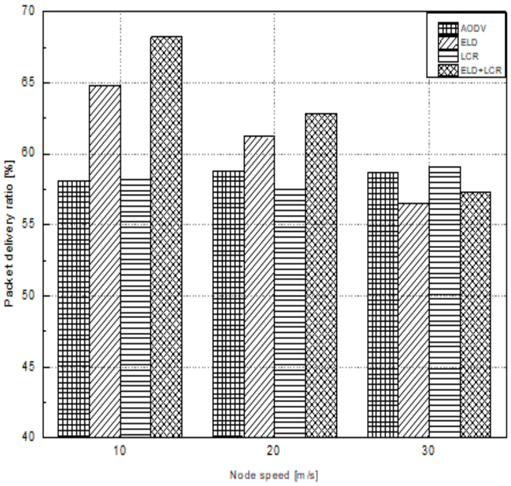

Fig. 5. Packet delivery rate in a network of 20 nodes as a function of varying mobility

2.1 Number of hello messages sent

From Fig. 4, we observe that the traditional approach almost exhibits a constant tendency with a slightly reduction in the number of Hello messages sent as node mobility increases. Meanwhile, other adaptive schemes react as expected by sending more messages as mobility increases. The combined approach (ELD+LCR) performs best by reducing overhead due to Hello messages by up to 43.8% compared to the traditional approach when conditions are less mobile at 10m/s and 28.62% when mobility is at the highest level. At low speed, the ELD parameter reacts by sending more messages compared to the LCR because regardless of node mobility at any given moment the minimum hold timer stored could be small. Also, at low speed, the chances that scattered nodes will stay empty for a longer period of time hence forced to use the default interval of 1 second.

2.2 Packet delivery ratio

Packet delivery rate depends on the ability of the applied scheme to detect changes and update links state information. Failure to detect these changes results in increased number of inconsistency routes or unidirectional links. From Fig. 5, we can see that the traditional approach and the LCR tend to exhibit almost similar tendencies by almost maintaining the rate of data packets delivered regardless of node mobility change. The ELD and the combined ELD+LCR performs best. However, the delivery rate tends to drop slightly as the network becomes less stable. The combined approach performs best by increasing the delivery rate by up to 16.6% at low mobility with a minor drop of just about 2% when mobility is high. Considering the reduction of overhead, we consider this slight drop in data delivery rate to be tolerable.

2.3 Normalized routing overhead

As previously mentioned, the normalized routing overhead measures the number of Hello messages required to keep link state fresh for a successfully transfer of a single data packet to the desired destination. It is clear that, the best scheme would be the one that would require less number of Hello messages to achieve and guarantee successful data transfer with the smallest amount of Hello messages possible. From Fig. 6 we observe that, the adaptive schemes reacts by sending more messages as mobility increases while the

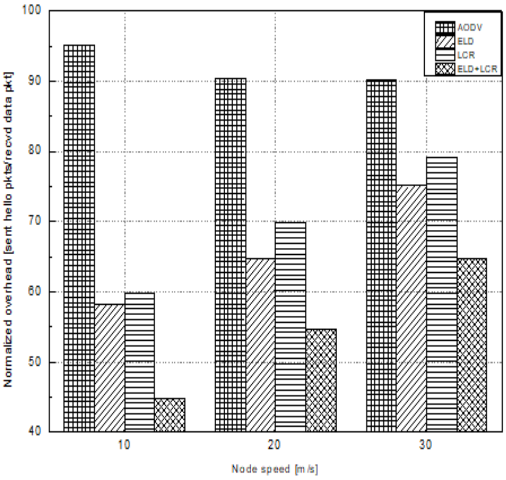

Fig. 6. Normalized routing overhead in a network of 20 mobile nodes as a function of varying mobility

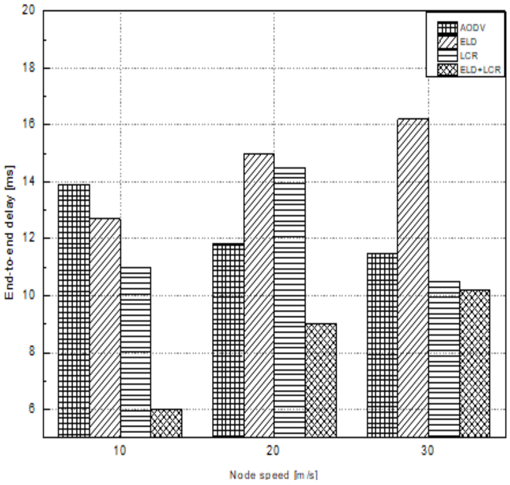

Fig. 7. End to end delay in a network of 20 mobile nodes as a function of varying mobility

original approach almost stays constant. As node mobility increases so do link failure rate. For that matter frequent updates become necessary to ensure successful delivery of data packets. The combined approach performs best compared to other schemes. Like in AODV, the results clearly show that, it is possible to maintain as well as improve the performance of the MANET routing protocols even with a fewer number of Hello messages released into the network.

2.4 End-to-end delay

Latency to deliver data packets depends on the ability of a scheme to update and discover correct link information especially in cases of a detected link failure. From Fig. 7, we observe that the adaptive timer scheme performance degrades as node speed increases ,unlike the traditional scheme. AODV reacts to link failure by first attempting the local repairing process and if failed the source repairing process is initiated, which means routed data are subjected to late delivery.

Amongst the adaptive schemes presented, the ELD mechanism tends exhibit worse performance, a fact caused by tendency of nodes to send Hello messages at huge interval differences. This behaviour subject nodes to fail or delay symmetric link connections even though they are within each other`s communication range.

Note, of all the presented approaches, the combined approach (ELD+LCR) exhibit a better performance compared to the rest by reducing the average delay of data packets by almost 56% when mobility is low and 8.6% when mobility is high compared to the original approach.

IV. Conclusions

In this paper, we discussed and presented various schemes to adaptively control the rate of sending Hello message. The effectiveness of each method is discussed and presented with the help of simulation results. From these results, we concluded that, a combination of the ELD and LCR parameters guarantee better performance in reducing the number of unnecessary Hello messages that would otherwise be released into the network. However further studies and experimentation on different network scenarios and other MANETs routing protocols are necessary to be done. Through this study, we concludes that, it is possible to improve or in some cases maintain the performance of MANET routing protocols with minimum overhead released into the network. Also, the timeout timer of neighbor nodes depending on neighbor Hello interval is not always the optimal way to track and update valid link state information. We expect to further extend our research and study the performance of our proposed schemes on other MANET routing protocols. In addition to that, we aim to study and apply decision making methods like reinforcement learning and multi-criteria decision making methods to adjust refresh timer intervals.

References

[1] M. Channappagoudar, P. Venkataram, “Performance Evaluation of Mobile Agent Based Resource Management Protocol for Manets,” Ad Hoc Networks, Vol. 36, No. 1, pp. 308-320, 2016.

[2] Y. Huang, N. Saleem, P. Daryl, “Tuning Olsr,” Proceedings of IEEE International Conference on Personal, Indoor and Mobile Radio Communications, pp. 1-5, 2006.

[3] R. Ernst, J. Sascha, B. Tobias, “Reducing Manet Neighborhood Discovery Overhead,” Proceedings of Annual IEEE Confererence on Local Computer Networks, pp. 374-377, 2014.

[4] S. Russell, N. Wang, D. Wright, “Performance Evaluation and Analysis of Proactive and Reactive Manet Protocols at Varied Speeds,” Proceedings of Annual Computing and Communication Workshop and Conference, pp. 981-985, 2020.

[5] N. Park, G. Kim, Y. Chuo, “Impact of Hello Interval and Link Change Ratio on Throughput in Manets,” Proceedings of International Conference on Ubiquitous and Future Networks, pp. 610-612, 2017.

[6] V. Giruka, M. Singhal, “Hello Protocols for Ad-hoc Networks: Overhead and Accuracy Tradeoffs,” Proceedings of International Conference on a World of Wireless Mobile and Multimedia Networks, pp. 354-361, 2005.

[7] Y. Huang, N. Saleem, S. Sørensen, “Reducing Neighbour Detection Latency in Olsr,” Proceedings of International Conference on Personal, Indoor and Mobile Radio Communications, pp. 1-5, 2007.

[8] G. Jisha, P. Samuel, “Impact of Node Density on Node Connectivity in Manet Routing Protocols,” Proceedings of International Conference on Advances in Computing and Communications, pp. 1-8, 2011.

[9] M. Benzaid, P. Minet, K. Agha, “Integrating Fast Mobility in the Olsr Routing Protocol,” Proceedings of International Workshop on Mobile and Wireless Communications Network, pp. 217-221, 2002.

[10] N. Hernandez, K. Shoji, T. Yutaka, “Dynamic Hello/timeout Timer Adjustment in Routing protocols for reducing overhead in manets,” Computer Communications, Vol. 33, No. 15, pp. 1864-1878, 2010.

[11] F. Jovel, J. McCartney, P. Teller, E. Ruiz, M. McGarry, “Neighbor Discovery Message Hold Times for MANETs,” Computer Communications, Vol. 112, pp. 38-46, 2017.

[12] The network simulator: ns-3. Available from: <http://www.nsnam.org>.

[13] N. Park, J. Neung, J. Nam, Y. Cho, “Impact of Node Speed and Transmission Range on the Hello Interval of Aodv Routing Protocol," Journal of Korean Institutes of Communications and Information Sciences, Vol. 43, No. 2, pp. 241-251, 2018.

[14] R. Colletti, M. Moraes, “Evaluation of Link Lifetime for the Random Waypoint Mobility Model,” Proceedings of International Conference on Networks, pp. 1-5, 2013.

[15] X. Wang, C. Wang, G. Cui, Q. Yang, “Practical Link Duration Prediction Model in Vehicular ad hoc Networks,” International Journal of Distributed Sensor Networks, Vol. 11, No. 3, 2015.

Daniel Godfrey

He received his M.S. degree from Chungnam National University, Korea. He is currently working toward Ph.D. degree at the same university.

Email: danielgodfrey2@gmail.com

Ki-Il Kim

He received the M.S. and Ph.D. degrees in computer science from the Chungnam National University, Daejeon, Korea, in 2002 and 2005, respectively.

Email: kikim@cnu.ac.kr